网格计算环境中可视化技术的研究*)

李金宝[1,2] 李建中[1,2] 商 超[2] 于 玲[2]
(哈尔滨工业大学计算机科学与工程学院 哈尔滨150001)[1]
(黑龙江大学计算机科学技术学院 哈尔滨150080)[2]

Visualization Research in the Grid Computing Environment

LJ Jin-Bao[1,2] LI Jian-Zhong[1,2] SHANG Chao[2] YU Ling[2]
(College of Computer Science and Engineering, Harbin Institute of Technology, Harbin 150001, China)[1]
(College of Computer Science and Technology, Heilongjiang University, Harbin 150080, China)[2]

Abstract Visualization research is one of the critical techniques in the Grid computing. In this paper a visual method is used to implement a distributed parallel computing program which makes the node point's status of the parallel system visible during the program's running time. At the end of the paper, the implementation of the visiual algorithm is given.

Keywords Visualization, Grid computing, Distributed parallel computing

最近几年随着计算机技术的发展,网络通信技术也得到了迅猛的发展,网络规模不断扩大,网络速度也在不断提高。特别是随着 Internet/Intranet 技术的兴起,网络应用几乎遍布了整个世界。在广域网或局域网中,微机或大型工作站的平均利用率只有30%左右,有的甚至不足10%。如何充分地利用这些地理上分布、异构的多种计算机(以下简称节点)资源组成网络虚拟超级计算机,来完成重大科学领域的大规模计算问题和其它复杂的任务,是分布式并行计算的重要研究内容[4]。网格计算的环境是由性能各异、分布、异构以及负载状态等不断变化的异构计算机资源组成,因此,任务调度和负载平衡等对整个系统的性能有重大影响[1]。

网格计算环境中的节点计算机往往具有不同的硬件平台及操作系统,在这样的环境下设计算法,需要考虑通信的延迟、网络拥塞、任务分配、负载平衡等因素的影响,将计算任务合理地分配到各个节点计算机。在程序运行时,如果能够实时获得网络通信信息、任务分配状况、节点的负载状态等信息,将对实现一个高效的算法具有十分重要的作用。但目前的分布式并行计算程序无论是采用动态调度算法,还是静态调度,程序一经运行,一般无法获知各个节点机的运行状态、网络通信状况等,因此很难根据运行时的各种信息对算法进行调整或改进。

本文通过引入可视化的方法[3],对网格计算环境中的各个节点机的状态进行实时监测,并以矩阵乘法为例子,实现了一个基于 CORBA 的可视化分布式并行计算程序。

1. 可视化分布式并行矩阵乘法模型

我们采用客户/服务器模型,来实现可视化分布式并行矩阵乘法,服务器由各个节点机来承担[1,2]。并行矩阵乘法采用棋盘划分方法,即将矩阵划分成若干个子矩阵,把矩阵间的运算转为多个子矩阵的运算[2]。客户机按照一定的规则,首先将子矩阵分配到各个节点机,然后各个节点机进行并行计算,并将各自的运算结果返回给客户机。客户机主要是进行任务分配、调度和结果回收,并形成最终结果,同时将矩阵运算过程用可视化的方法进行描述,使用户可以直观了解并行程序执行的情况。

假设客户机和节点机上的时间都是一致的。设客户机发送第 i 个数据(可以是一个子阵)的时刻是 CS(i),某个接收该数据的节点机接收到该数据的时刻是 PR(i),该节点机对该数据完成处理后的时刻是 PC(i),客户机接收到该处理结果的时刻是 CR(i),把发送第 i 块数据时,服务器到节点机发送数据所用的通讯时间记为 CTP(i),节点机到服务器回传数据所用的通讯时间记为 PTC(i),则有:

$$CTP(i)=PR(i)-CS(i)$$

$$PTC(i)=CR(i)-PS(i)$$

设 CTP 为客户机向节点机发送数据所用的总通讯时间,PTC 为节点机到服务器回传数据所用的总的通讯时间,总数据块数为 n,则有

$$CTP=\sum_{i=1}^{n}CTP(i)=\sum_{i=1}^{n}(PR(i)-CS(i))$$

$$PTC=\sum_{i=1}^{n}PTC(i)=\sum_{i=1}^{n}(CR(i)-PS(i))$$

由于本文的可视化分布式并行计算是基于客户/服务器结构的,因此下面的算法分成客户端算法和服务器端(节点机)算法两个部分。

2. 可视化分布式并行矩阵乘法的节点机算法

2.1 节点机上矩阵乘法 mutiple 的实现

在服务器端,算法比较简单,由于我们采用的是简单分块算法实现并行与分布式矩阵乘法计算,因此,在服务器端实际

*)国家863计划CIMS专题资助项目(项目编号2001AA415410);黑龙江省教育厅科技资金资助项目。李金宝 副教授,博士生,主要研究领域为并行计算,计算机体系结构,数据库。李建中 教授,博士生导师,主要研究领域为数据库,数据仓库,并行计算。商 超 硕士研究生,主要研究领域为并行计算,网格计算,数据库。于 玲 硕士研究生,主要研究领域为并行计算,移动计算,数据库。

上就是一个单机上进行矩阵乘法运算的程序，这个程序以CORBA 组件的形式存在于节点机上，具体算法如下。

算法1

输　入：matr1 --- 双精度二维数组(用一维数组表示)；
　　matr2 --- 双精度二维数组(用一维数组表示)；
　　m--- 长整型，matr1的行数；
　　n--- 长整型，matr1的列数；
　　k--- 长整型，matr2的列数；

输　出：pointbegin --- 双精度，节点开始计算的时刻；
　　pointend --- 双精度，节点结束计算的时刻；
　　matr3--- 双精度的二维数组(用一维数组表示)；

算法描述：

```
pointbegin=开始计算的时刻;
FOR i 从0到 m-1
For j 从0到 k-1
    FOR t 从0到 n-1
          sum+=matr1[i*n+t]*matr2[t*k+j];
    matr3[i*k+j]=sum;
    sum=0;
pointend=结束计算的时刻;
```

3. 客户端程序实现

客户端的主要任务是调用各个节点机上的计算服务程序，实现矩阵的分布式计算。同时将矩阵运算过程用可视化的方法进行描述，使用户了解并行程序执行的情况。

3.1　可视化界面设计

客户程序的界面包含三个视图，它们的主要功能如下：

1. 显示并行计算过程的文本信息，使用户了解并行计算过程中客户机和各个节点机当前状态等信息。

2. 显示参与运算的机器信息。

3. 显示运算过程，主要包括客户机发送数据、节点机对数据计算、节点机回传运算结果、客户机数据集成四个过程。

可视化界面的设计中主要考虑并行计算信息的可视化反馈，这涉及到可视化描述过程与分布式计算过程的同步，后文将给出该同步算法。程序中采用不同颜色的进度条实现可视化进程的反馈，同时将参与运算的各个节点机信息保存到文件中。

3.2　客户端算法的实现

对于矩阵乘法，我们构造一种表结构，用于存储两个相乘子矩阵在原来相乘的两个矩阵中的位置。通过读取这种表结构可以获得相乘子矩阵的位置。对于矩阵乘法，需要存储矩阵一的行、列位置，和矩阵二的列位置，因此，定义如下表结构：

```
struct {
        int Mi;
        int Ni;
        int Ki;
      } Table;
```

定义上述表结构，可以为子矩阵的分配提供便利条件。在矩阵初始化的同时对表结构进行初始化。客户端算法采用多线程技术，各个线程通过互斥读取 Table 表获得各个子矩阵的位置，通过表中的成员变量获得子矩阵中的数据。

在程序的实现过程中定义了线程类 Mythread，该类中主要包括 readTable()，readSubMatrix()和 integrateResult()三个成员函数。下面给出这三个函数的算法和功能：

算法2　readTable()

功　能：读取可相乘的两个子矩阵；

输　入：&pmi，&pni，&pki 表示子阵相对原矩阵的位置。

返回值：0(表为空)，1(表非空)；

算法描述：

```
设置临界区标记;
if (TableLength>0){
    *pmi=Table[TableLength-1].Mi;
    *pni=Table[TableLength-1].Ni;
    *pki=Table[TableLength-1].Ki;
   TableLength--;
   if(tableLength)
         a=1;
   else
         a=0;
   解除临界区标记;
   return a;
    }
    else{
         解除临界区标记;
         return 0;
    }
```

算法3　readSubMatrix()

功　能：读取子矩阵的数据，并将数据存到 PsubMatrixA，PsubMatrixB 所指定的存储单元中。

输　入：mi，ni，ki 表示子阵在原矩阵的位置。PsubMatrixA，psubMatrixB 为指向 matrixtype 类型的指针。

算法描述：

```
subm=A 子阵的行数;
subn=A 子阵的列数;
subk=B 子阵的列数;
for i=0 to subm-1
    for j=0 to subn-1
      (*psubMatrixA)[i*subn+j]=
                    (*pMatrixA)[(subm*mi+i)*n+
                    (subn*ni+j)];
        for i=0 to subn-1
            for j=0 to subk-1
              (*psubMatrixB)[i*subk+j]=
                          (*pMatrixB)[(subn*ni+i)*k
                          +(subk*ki+j)];
```

算法4　integrateResult()

功　能：合成最终计算结果。

输　入：mi，ki 表示两个子阵计算的结果在结果矩阵的位置，psubMatrixC 为 matrixtype 类型的指针。

算法描述：

```
subm=子阵的行数;
subk=子阵的列数;
for i=0 to subm-1
    for j=0 to subk-1
          (*pMatrixC)[(subm*mi+i)*k+(subk*ki+j)]
                    +=(*psubMatrixC)[i*subk+j];
```

3.3　界面线程与计算线程的同步

可视化分布式并行计算的一个主要问题是同步问题，即计算线程与界面线程的同步。为了使计算的过程能用可视化的方法表示出来，必须对界面线程与计算线程加以控制，本文中通过使用共享公用变量和 CEvent 对象的方法来实现界面线程与计算线程的同步。下面给出计算线程与界面线程的算法：

算法5　计算线程 i 的(描述)算法

1)判断 Table 表中是否有尚未被读出的子阵信息，如果没有则结束；

2)设置计算线程 i 的线程结束标记为 FALSE；(Finish[i]=FALSE)

3)调用 readSubMatrix()函数读取分块子阵；

4)记录分发子阵的开始时刻，调用远程方法将(3)中读取的子阵分配到节点机上进行运算；(将与计算相关的进度条的标志位置为 TRUE)

5)等待，直到得到节点机上回传的运算结果，将与计算相关的进度条标志位置 FALSE，并记录收到数据的时刻；

6)互斥向文件中写入相关信息；

(下转第18页)

1994.903
24 Bussmann S. Agent-Oriented Programming of Manufacturing Control Tasks. In:Proc. of the 3rd Intl. Conf. on Multi-Agent Systems (ICMAS'98), Paris, France, 1998. 57～63
25 Butler J,Ohtsubo H. ADDYMS: architecture for distributed dynamic manufacturing scheduling. Artificial Intelligence Application in Manufacturing, MIT Press, 1992. 199～214
26 Sacile R. Telemedicine Systems for Collaborative Diagnosis over the Internet: Towards virtual "collaborators". In:Proc. of the Acadma/Industry working Conf. on Research Challenges (AIWORC'00), April 27-29, Buffalo, New York, 2000
27 Huang J,Jennings N R,Fox J. An agent architecture for distributed medical care. In:M. J. Wooldridge,N. R. Jennings, eds. Intelligent Agents lecture notes in Artificial Intelligence, Vol. 890, 1995. 219～232
28 Wangermann J P,Stengel R F. Principal negotiation between intelligent agents: a model for air traffic management. Artificial Intelligence in Engineering,1998,12:177～187
29 刘贵全,陈小平. 一个基于 Agent 的协作式学习系统. 中国科学技术大学学报,2000,30(1):113～118

(上接第7页)

7)调用 integrateResult()函数集成结果;

8)将与集成结果相关的进度条标志位置为 FALSE;

9)置 Finish [i]为 TRUE,等待界面线程 i 结束标记置为 TRUE;

10)判断 Table 中是否还有没有读完的数据块,如果有,转移到2;

如果没有,Again [i]置为 FALSE, Finish [i]置标志位;

算法6 界面线程 i 的(描述)算法

1)判断 Again [i]的初值是否为 TRUE,如果不为 TRUE,则结束。

2)Finish [i]置为 FALSE。Continue[i][1]置为 TRUE, continue[i][3]置为 TRUE。

3)用进度条显示数据发送的过程。

4)如果 Continue[i][1]为 FALSE,转移到6)。

5)与 Continue[i][1]对应的进度条开始反馈运行信息,转移到4)。

6)与 Continue[i][1]对应的进度条显示完毕。

7)用进度条显示接收数据的过程。

8)如果 Continue[i][3]不为 TRUE,则转移到10)。

9)与 Continue[i][3]对应的进度条开始反馈信息,转移到8)。

10)与 Continue[i][3]对应的进度条显示结束。与本线程对应的线程结束标记置 TURE,等待 Finish [i]标记置 TRUE,转到1)。

4. 可视化分布式并行矩阵乘法的实现

本文的可视化分布式并行计算模型已在一个由9台异构 PC 机组成的局域网上实现,操作系统使用 Windows 2000 Server,CORBA 采用 IONA 公司的 ORBacus 4.1.3[6～8],编程语言使用 C++。

实验采用的机器如下:5台128M 内存 PIII800机器,2台256M 内存 PIII500机器,2台128M 内存 AMD 300机器。其中一台 PIII800机器做为客户机,其它机器全部用作节点机。图1是在三个节点的情况下,维数为512的两个方阵相乘时的可视化程序的运行情况。

从图1中可以看到矩阵的整个运算过程和各节点机的运行状态。如节点2、节点3正处于计算状态;节点1已经完成了一个计算任务,并已将计算结果传给客户机,此时客户机正在集成节点1回传的结果,同时节点1正准备接收新的计算任务。由此可以看出,可视化技术可以使用户了解计算节点在某一时刻的状态。

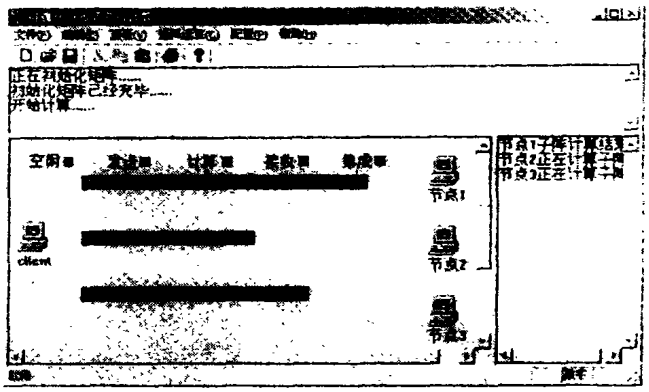

图1

结束语 在分布式并行计算中引入可视化的方法,可以将整个计算的过程直观地呈现给用户,使用户了解计算节点在某一时刻的状态,便于用户对其所设计的并行算法进行分析、调整和评价,并在此基础上设计出效率更高的算法。同时可视化技术还可以使用户方便地进行并行程序的调试,并在一定程度上降低并行算法的设计和调试难度。

参 考 文 献

1 Aleksy M, Korthaus A. A CORBA-Based Object Group Service and a Join Service Providing a Transparent Solution for Parallel Programming. 2000 IEEE
2 陈国良. 并行计算. 高等教育出版社
3 (美)Hwang K 著,王鼎兴等译. 高等计算机系统结构. 清华大学出版社
4 肖侬,等. 面向21世纪的广域高性能元计算技术. 计算机科学, 1999, 26(10)
5 王育峰,杨寿保. 网格计算系统研究及发展方向. 计算机科学, 2002,29(6)
6 CORBA/C++ Programming with ORBacus. http: // www. ooc. com
7 Jthreads/C++ (Java-like Threads for C++). http: // www. ooc. com
8 Henning M,Vinoski S. 基于C++的CORBA 高级编程. 清华大学出版社,2000. 7